A

**SCE / PBL**

On

**“(IPL Data Analysis)”**



**VISHWAKARMA INSTITUTE OF**

**INFORMATION TECHNOLOGY, PUNE**

**INFORMATION TECHNOLOGY DEPARTMENT**

**BY:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr No.** | **Roll no.** | **Name** | **PRN** |
| 1 | 231027 | Sagar Jadhav | 22010365 |
| 2 | 231032 | Satyam kangle | 22010972 |

**Division: A Sem -IV Batch: A-2**

**Sub: ITW-Python**

**Introduction :-**

The use of analytical methods in various aspects of cricket such as Batting, Bowling, Fielding, Team Selection, Result Prediction, Target Revision in a rain affected match are very important. There is a huge demand of various algorithms for each and every aspect of cricket because of its popularity and the huge amounts of money involved in the game.

In India the followers of cricket are also followers of statistical records. Thus, the analysis of a league like IPL becomes more important .

Prediction of outcome of matches using algorithm approaches is one of the very important aspects in cricket. No comprehensive attempt has been made in the literature to this end. This is an important problem because franchises invest huge amounts of money.

With the help of a coin toss one can predict with 50% probability the winning team treating the game as one of perfect chance without even considering the relative merits of the respective teams playing against each other. In contrast to this records of past performance of players, their current form and other cricket related data can be analyzed statistically to create mathematical models with better probability of success in predicting the winning team.

**Data description:**

In this project we use different type of the parameter related to IPL team’s information

Name of players, Team, Nationality, Player Type, Capped, Matches Played, Runs Average, Strike Rate, Wickets, Bowling average, Economy, Bowling Strike Rate, Catches, Run outs, Stump outs.

**Code:**

import pandas as pd

import numpy as np

import seaborn as sns

import matplotlib.pyplot as plt

%matplotlib inline

import warnings

warnings.filterwarnings('ignore')

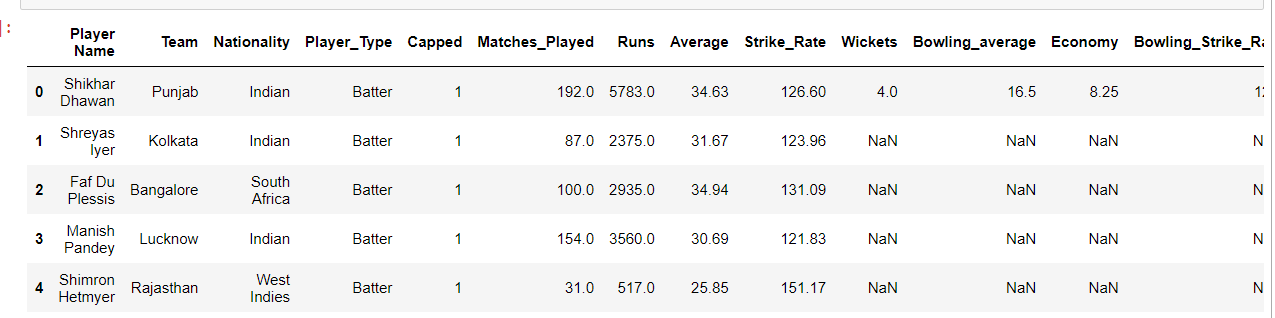
ipl=pd.read\_csv('IPLData.csv')

ipl1=pd.read\_csv('IPLData1.csv')

# prints the first five lines of code

ipl1.head()

**Output:**

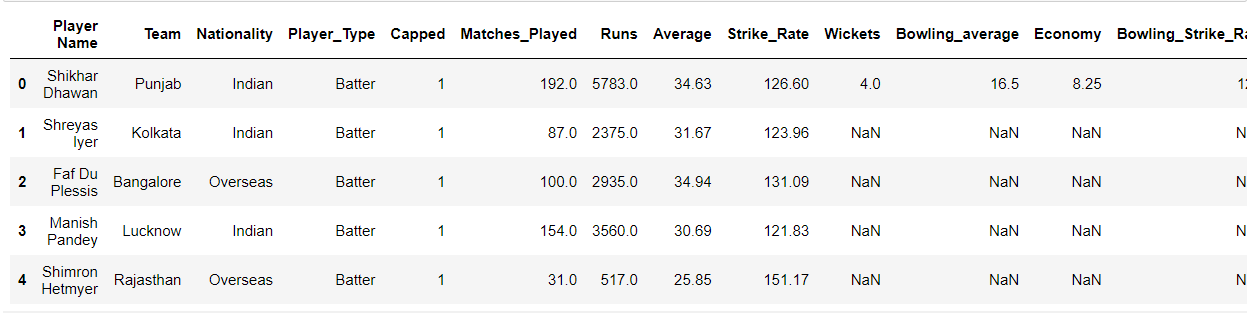
****

**Code:**

# display the first five lines

ipl.head()

**Output:**

****

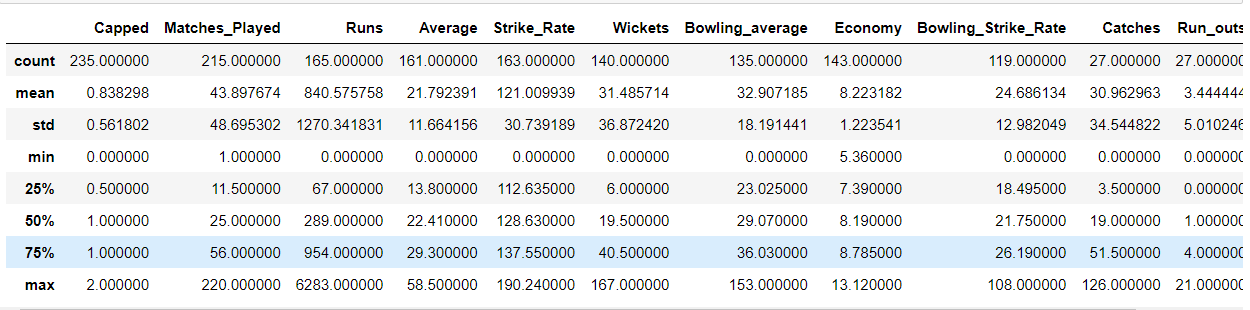
**Code:**

import pandas as pd

# Gives the information about following parameters

ipl.describe()

**Output:**

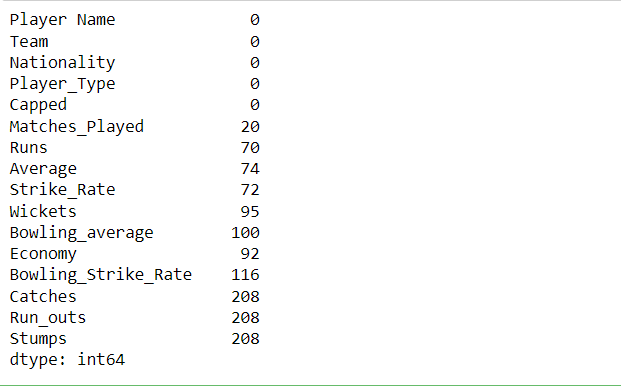
****

**Code:**

#check the number of null values in each column

ipl.isna().sum()

**Output:**

****

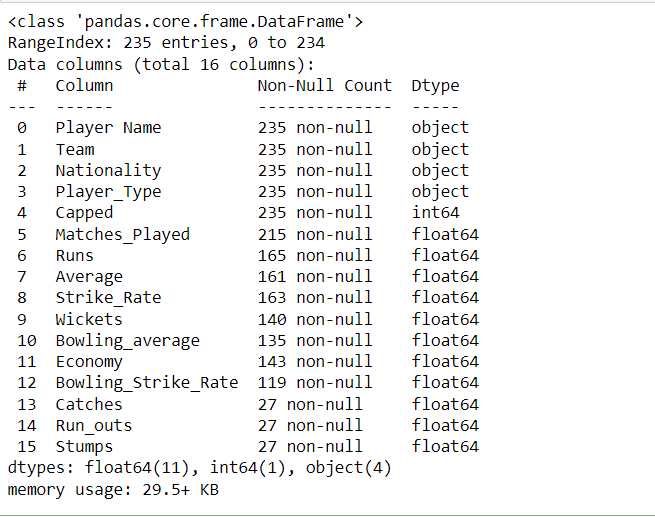
**Code:**

# no of non-nullvalues

ipl.info()

#retun information about the data set

**Output:**

****

**Code:**

#1.Capped players- batters,bowlers,all-rounder,wicket-keeper

#2.Uncapped players- batters,bowlers,all-rounder,wicket-keeper

# seperating capped batters

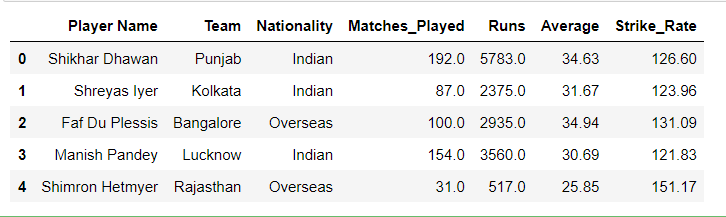
batters = ipl.loc[(ipl["Player\_Type"] == "Batter")]

batters\_new = batters.loc[(batters["Capped"] == 1)]

Capped\_Batters = batters\_new[['Player Name','Team','Nationality','Matches\_Played','Runs','Average','Strike\_Rate']]

Capped\_Batters.head()

**Output:**

****

**Code:**

# capped keepers

keepers = ipl.loc[(ipl["Player\_Type"] == "Keeper")]

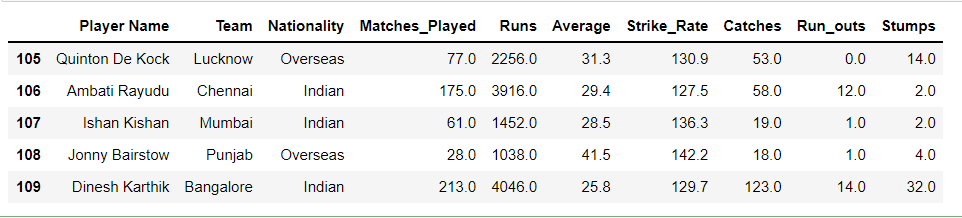
keepers\_new = keepers.loc[(keepers["Capped"] == 1)]

Capped\_keepers = keepers\_new[['Player Name','Team','Nationality','Matches\_Played','Runs','Average','Strike\_Rate','Catches',

'Run\_outs','Stumps']]

Capped\_keepers.head()

**Output:**

****

**Code:**

# capped all-rounders

Allrounders = ipl.loc[(ipl["Player\_Type"] == "Allrounder")]

Allrounders\_new = Allrounders.loc[(Allrounders["Capped"] == 1)]

Capped\_Allrounders = Allrounders\_new[['Player Name',

'Team',

'Nationality',

'Matches\_Played',

'Runs',

'Average',

'Strike\_Rate',

'Wickets',

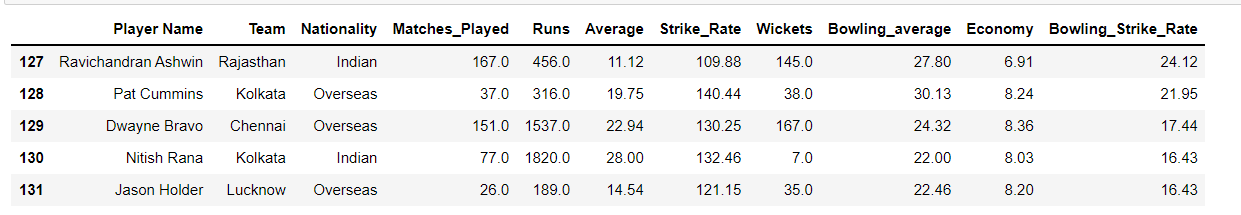
'Bowling\_average',

'Economy',

'Bowling\_Strike\_Rate']]

Capped\_Allrounders.head()

**Output:**

****

**Code:**

import pandas as pd

#cleaning the data by making null value by 0

Capped\_Batters = Capped\_Batters.fillna(0)

Capped\_Bowlers = Capped\_Bowlers.fillna(0)

Capped\_Allrounders = Capped\_Allrounders.fillna(0)

Capped\_keepers = Capped\_keepers.fillna(0)

#checking null values in data

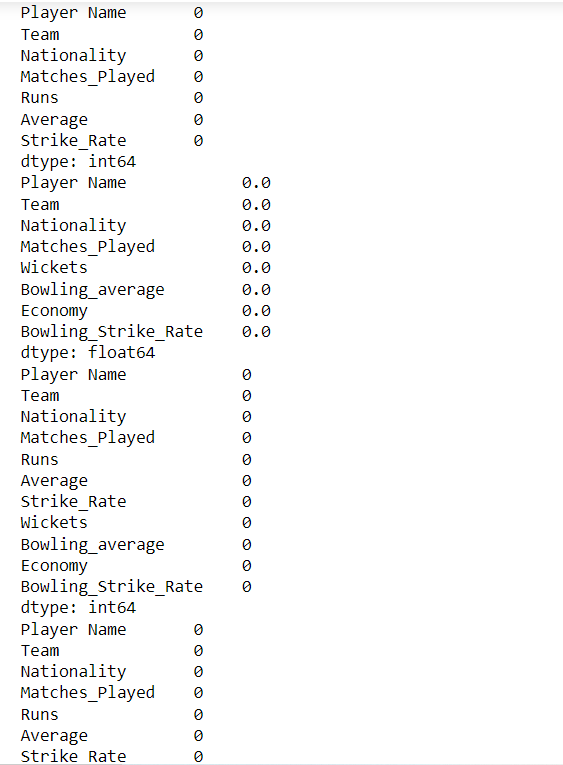
print(Capped\_Batters.isna().sum())

print(Capped\_Bowlers.isna().sum())

print(Capped\_Allrounders.isna().sum())

print(Capped\_keepers.isna().sum())

**Output:**

****

**Code:**

# analyzing the batters data

#batting average more than 32.0

top\_batters = Capped\_Batters.loc[(Capped\_Batters["Average"] >= 32.0)]

#sorting the in descending order

top\_batters\_average = top\_batters.sort\_values('Average',ascending=False)

top\_batters\_strike\_rate = top\_batters.sort\_values('Strike\_Rate',ascending=False)

top\_batters\_runs = top\_batters.sort\_values('Runs',ascending=False)

top\_batters\_matches = top\_batters.sort\_values('Matches\_Played',ascending=False)

# data of each of the batters in descending order

top\_batters\_average

# strike rate in descending order

top\_batters\_strike\_rate

#runs scored in descending order

top\_batters\_runs

#matches played

top\_batters\_matches

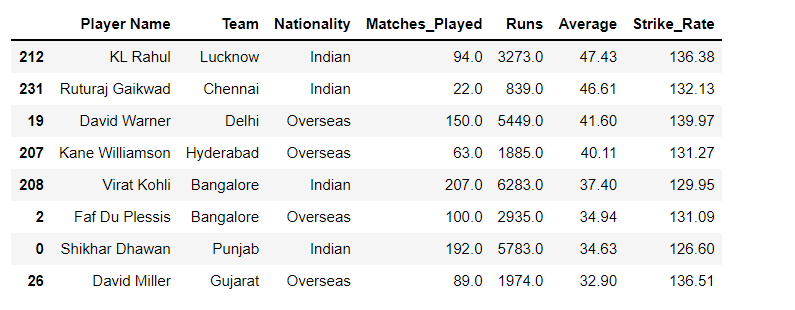
**#from our analysis,if we rank 0 to 10, the top three batters that will come while analysiseach of the above data are:**

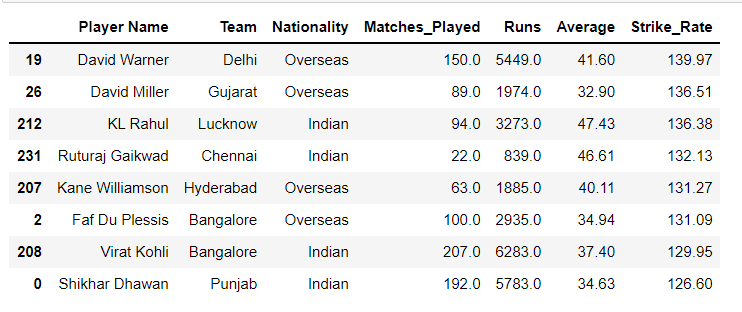
**# 1.David Warner**

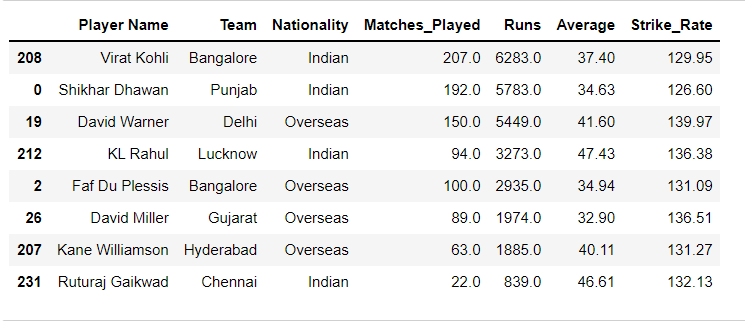
**# 2.KL Rahul**

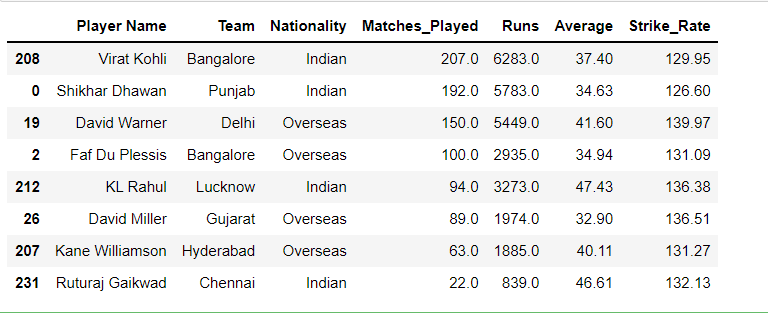
**# 3.Virat Kohli**

**Output:**

****

****

****

****

**Code:**

import pandas as pd

#Analyzing the Allrounder Data

#we have marrowed our analysis by further sagregating the allrounders based on strike rate equal to or more than 148.8.

top\_allrounders = Capped\_Allrounders.loc[(Capped\_Allrounders["Strike\_Rate"] >= 140.0)]

top\_allrounders\_average = top\_allrounders.sort\_values('Average', ascending=False)

top\_allrounders\_strike\_rate = top\_allrounders.sort\_values('Strike\_Rate', ascending=False)

top\_allrounders\_runs = top\_allrounders.sort\_values('Runs', ascending=False)

top\_allrounders\_matches = top\_allrounders.sort\_values('Matches\_Played', ascending=False)

top\_allrounders\_bowling\_average = top\_allrounders.sort\_values('Bowling\_average')

top\_allrounders\_bowling\_strike\_rate = top\_allrounders.sort\_values('Bowling\_Strike\_Rate')

top\_allrounders\_wickets = top\_allrounders.sort\_values('Wickets', ascending=False)

top\_allrounders\_economy = top\_allrounders.sort\_values('Economy')

top\_allrounders\_average

top\_allrounders\_strike\_rate

top\_allrounders\_runs

top\_allrounders\_matches

top\_allrounders\_bowling\_average

top\_allrounders\_bowling\_strike\_rate

top\_allrounders\_wickets

top\_allrounders\_economy

**# rank for top allrounders based on above analysis**

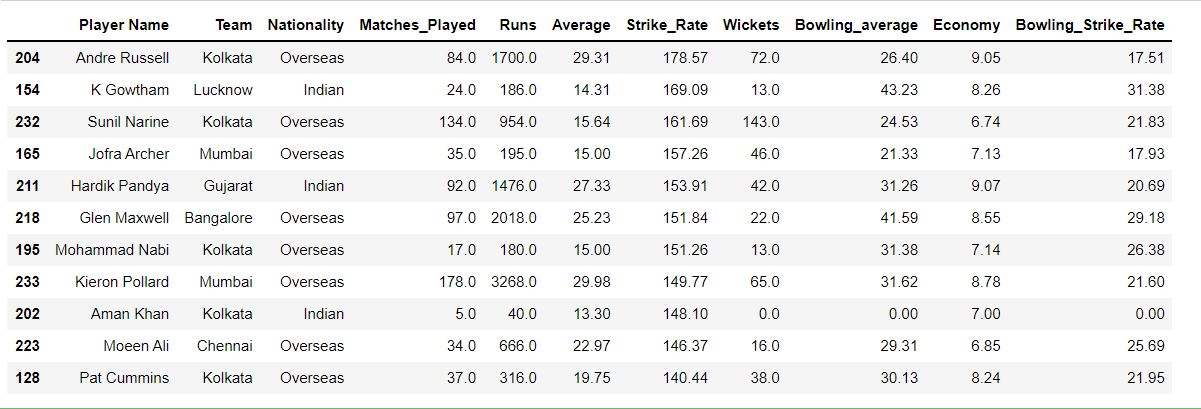
**#1. Andre Russsell**

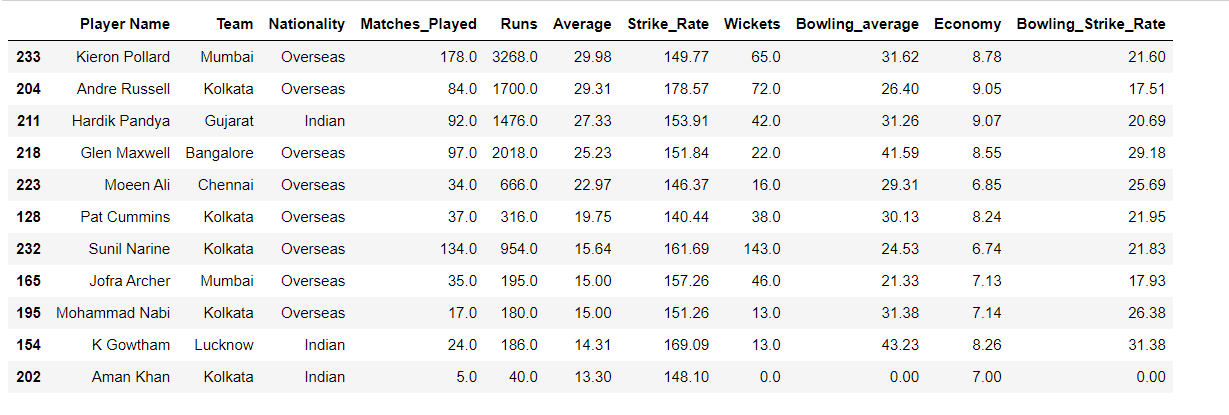
**#2. Sunil Narine**

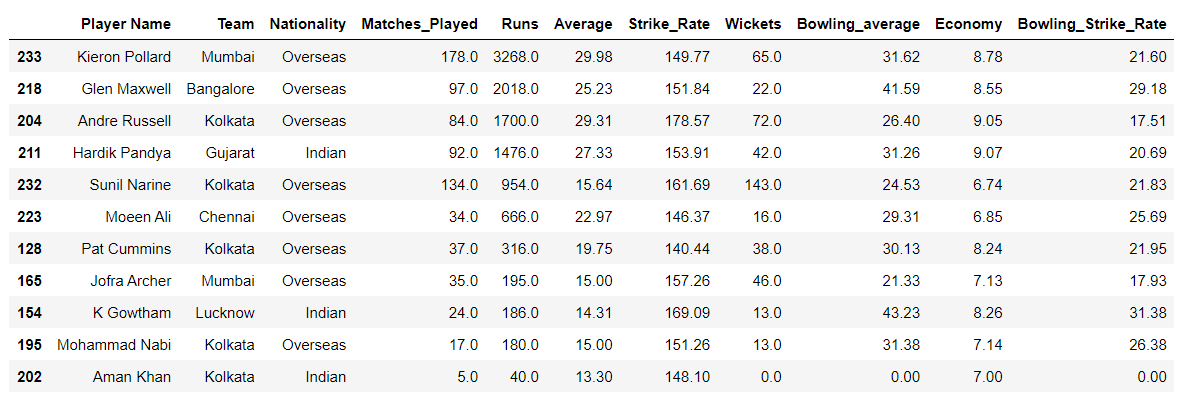
**#3. Hardik Pandya**

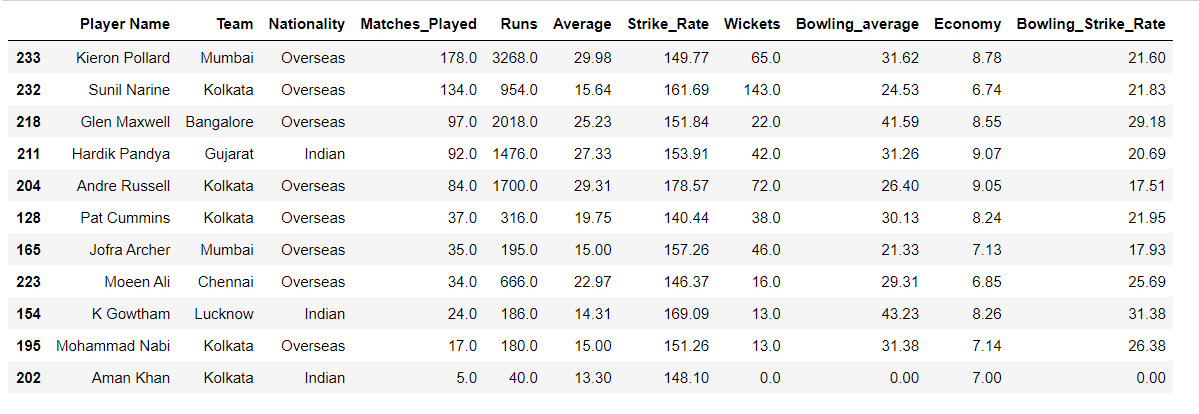
**#4. Jofra Archer**

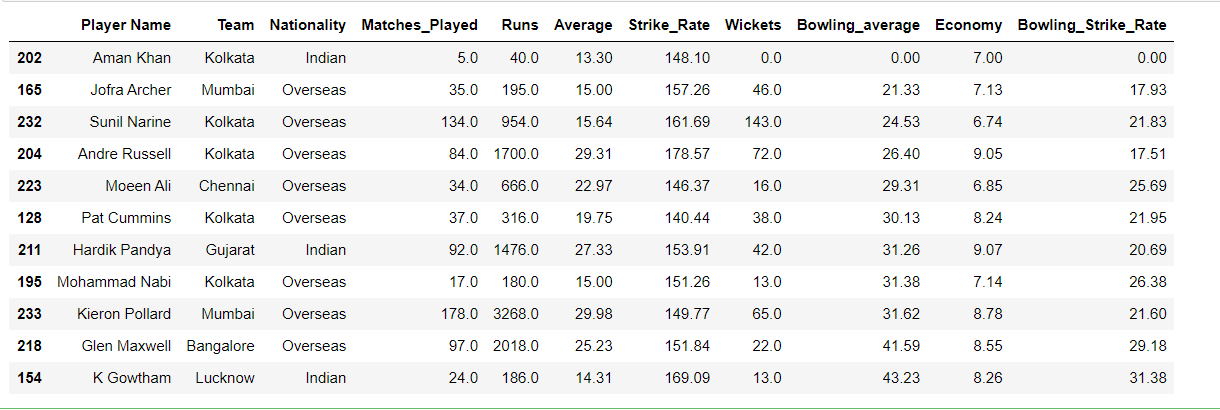
**Outputs:**

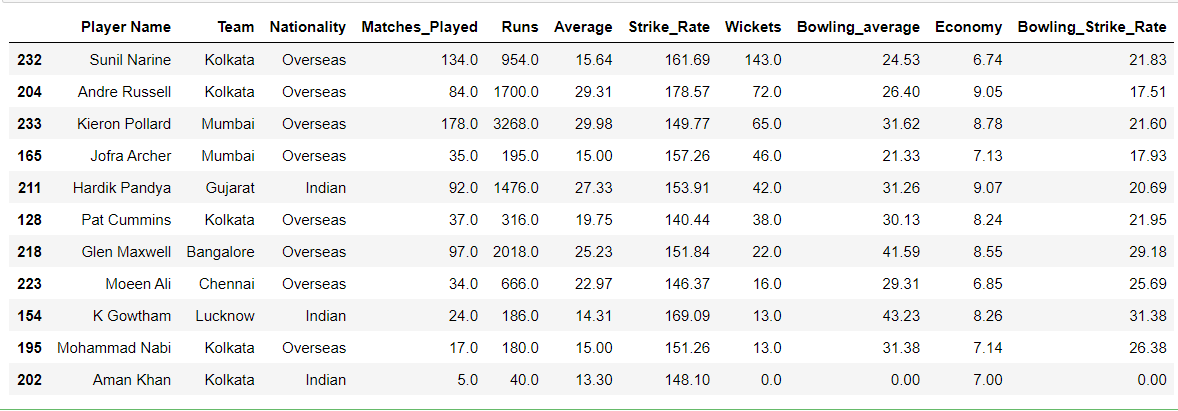
****

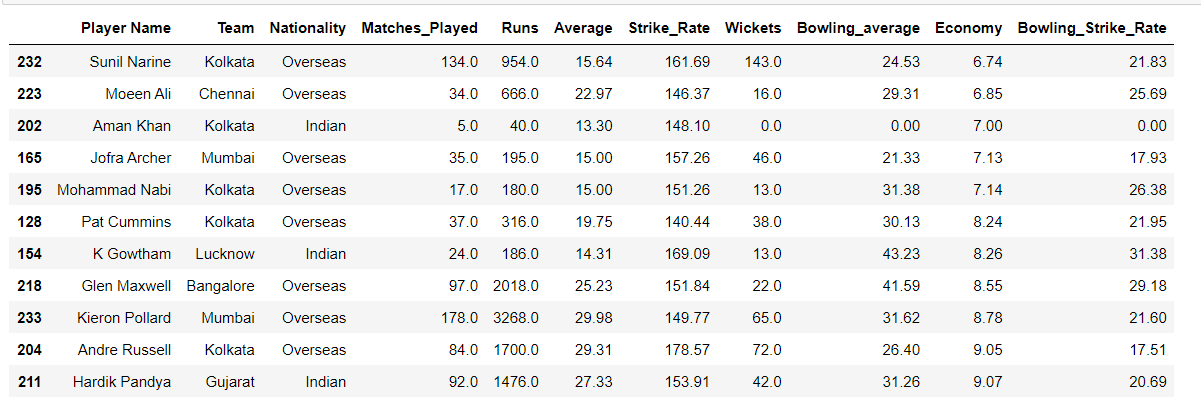
****

****

****

****

****

****

**Code:**

#Analyzing the Keepers Data

#we have narrowed our analysis by further sagregating the keepers based on avarage more than 25.0.

top\_keepers = Capped\_keepers.loc[(Capped\_keepers["Average"] >= 25.0)]

top\_Keepers\_average = top\_keepers.sort\_values('Average', ascending=False)

top\_Keepers\_strike\_rate = top\_keepers.sort\_values('Strike\_Rate', ascending=False)

top\_Keepers\_runs = top\_keepers.sort\_values('Runs', ascending=False)

top\_Keepers\_matches = top\_keepers.sort\_values('Matches\_Played', ascending=False)

top\_Keepers\_catches = top\_keepers.sort\_values('Catches',ascending=False)

top\_Keepers\_runouts = top\_keepers.sort\_values('Run\_outs',ascending=False)

top\_Keepers\_stumps = top\_keepers.sort\_values('Stumps', ascending=False)

top\_Keepers\_average

top\_Keepers\_strike\_rate

top\_Keepers\_runs

top\_Keepers\_matches

top\_Keepers\_catches

top\_Keepers\_runouts

top\_Keepers\_stumps

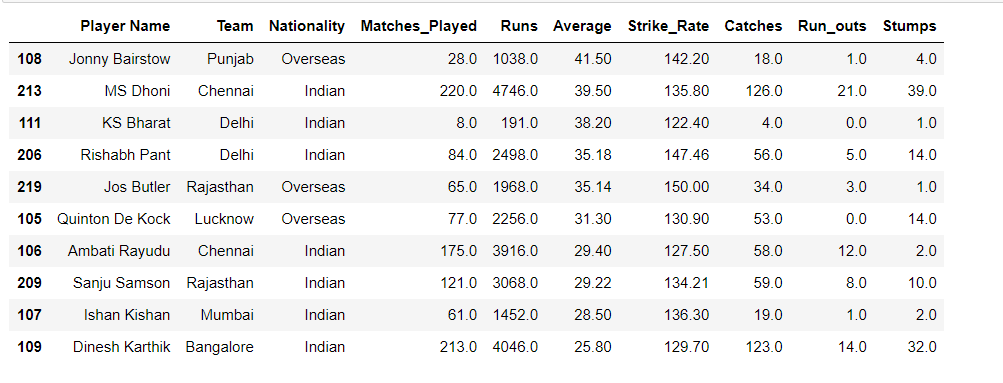
**# Rank for keepers from above data**

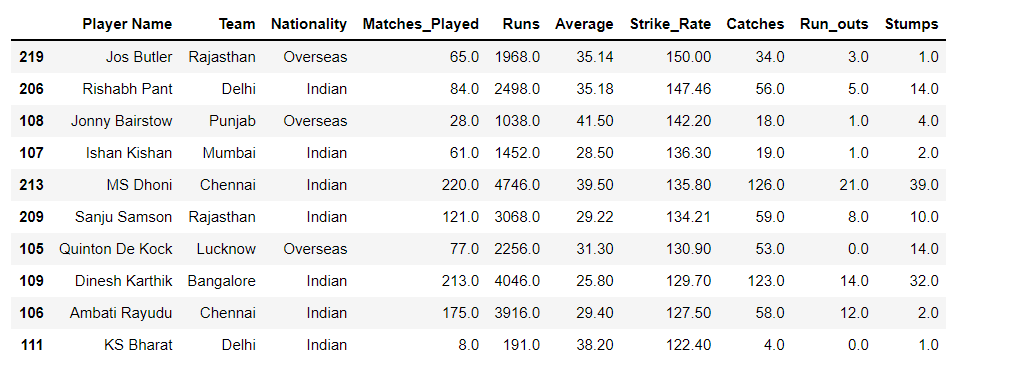
**# 1.MS Dhoni**

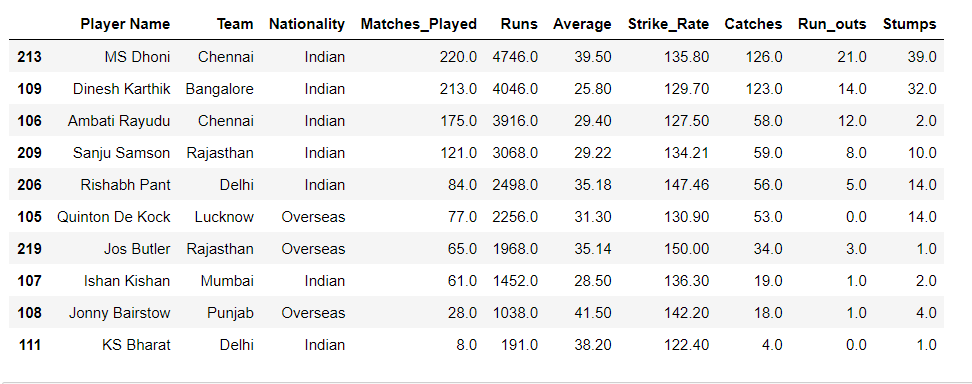
**# 2.Dinesh Karthik**

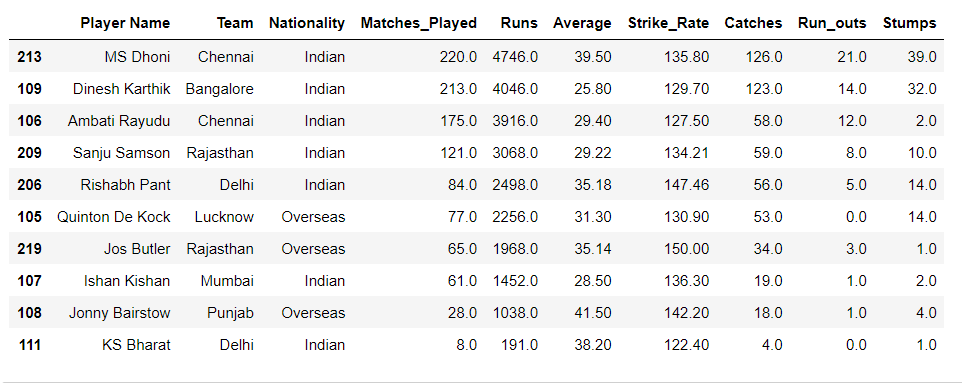
**# 3.Rishabh Pant**

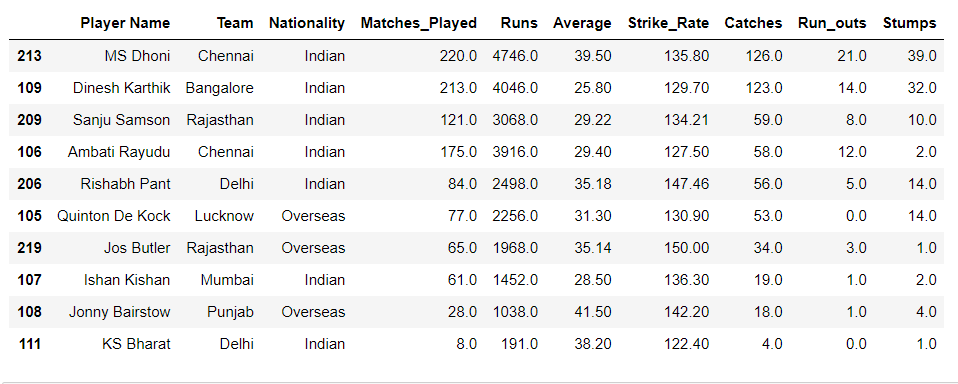
**Outputs:**

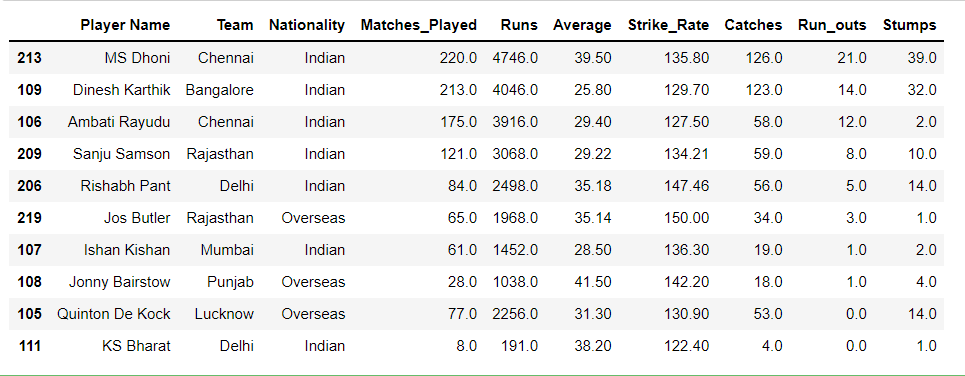
****

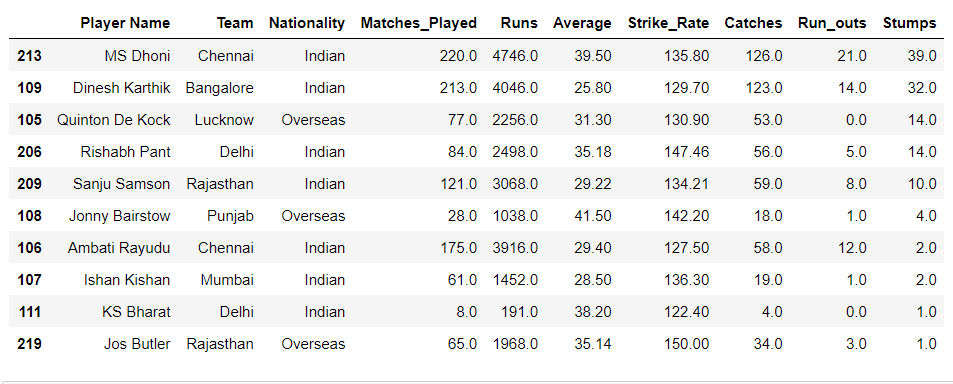
****

****

****

****

****

****

**Code:**

#How many types of player have participated?

types = ipl['Player\_Type'].value\_counts()

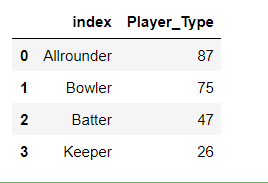
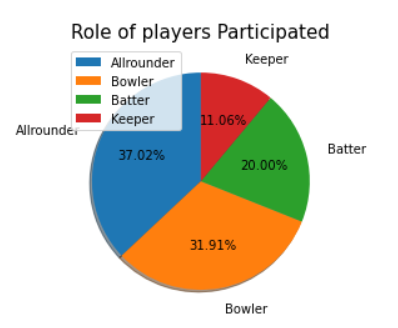
types.reset\_index()

plt.pie(types.values, labels=types.index,labeldistance=1.2,autopct='%1.2f%%',shadow=True,startangle=60)

plt.title('Role of players Participated', fontsize =15)

plt.plot()

**Output:**

** **

**Code:**

plt.figure(figsize=(10,5))

#palette=['Orange','Pink','Red']

fig= sns.countplot(ipl['Capped'])

plt.xlabel('Capped or Uncapped')

plt.ylabel('No. of player')

plt.title('Capped or Uncapped',fontsize=15)

plt.plot()

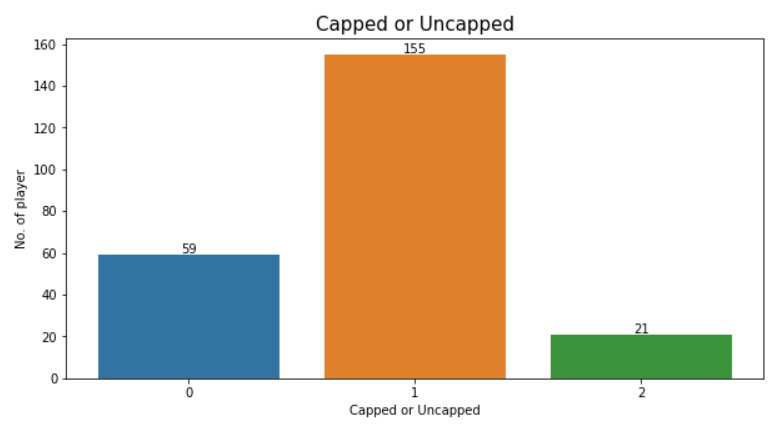
for p in fig.patches:

fig. annotate(format (p.get\_height(), '.0f'), (p.get\_x() +

p.get\_width()/2., p.get\_height()), ha= 'center', va = 'center',

xytext = (0, 4), textcoords= 'offset points')

**Output:**

****

**Code:**

#How many types of countries player have participated?

types = ipl1['Nationality'].value\_counts()

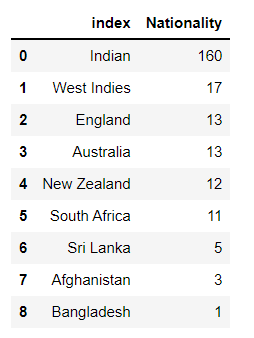
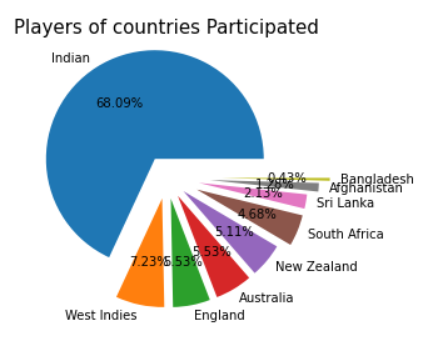
types.reset\_index()

plt.pie(types.values, labels=types.index,labeldistance=1.2,autopct='%1.2f%%',shadow=True,startangle=60)

plt.title('Players of countries Participated', fontsize =15)

plt.plot()

**Output:**

** **

**Code:**

# Total number of players bought by each team.

plt.figure(figsize=(20,10))

fig = sns.countplot (ipl[ipl[ 'Team']!= 'Unsold' ][ 'Team'])

plt.xlabel('Team Names')

plt.ylabel('Number of Players')

plt.title('Players Bought by Each Team', fontsize=12)

plt.xticks (rotation=70)

plt.plot()

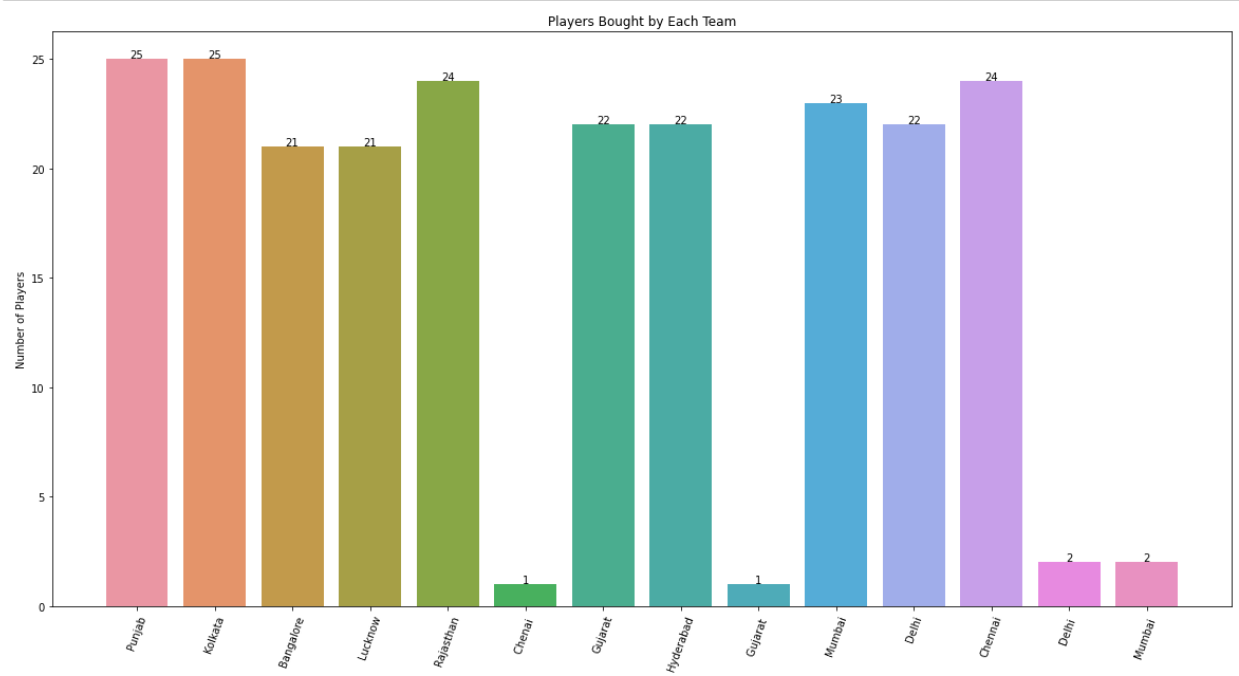
for p in fig.patches:

fig. annotate(format (p.get\_height(), '.0f'), (p.get\_x() +

p.get\_width()/2., p.get\_height()), ha = 'center', va = 'center', xytext = (0, 4),

textcoords='offset points')

**Output:**

****

**Code:**

#visualization of batters data

#PLot Shows each of the top batters.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Strike\_Rate', data = top\_batters)

#visualization of batters data

#PLot Shows each of the top batters.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Runs', data = top\_batters)

#visualization of batters data

#PLot Shows each of the top batters.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Average', data = top\_batters)

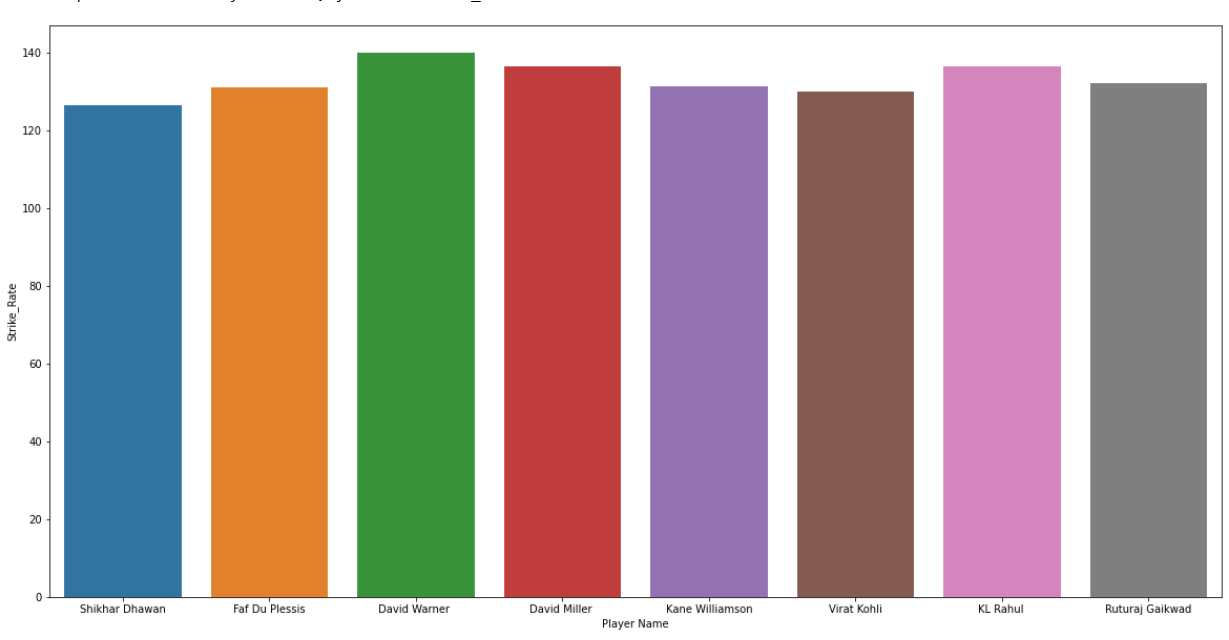
#visualization of batters data

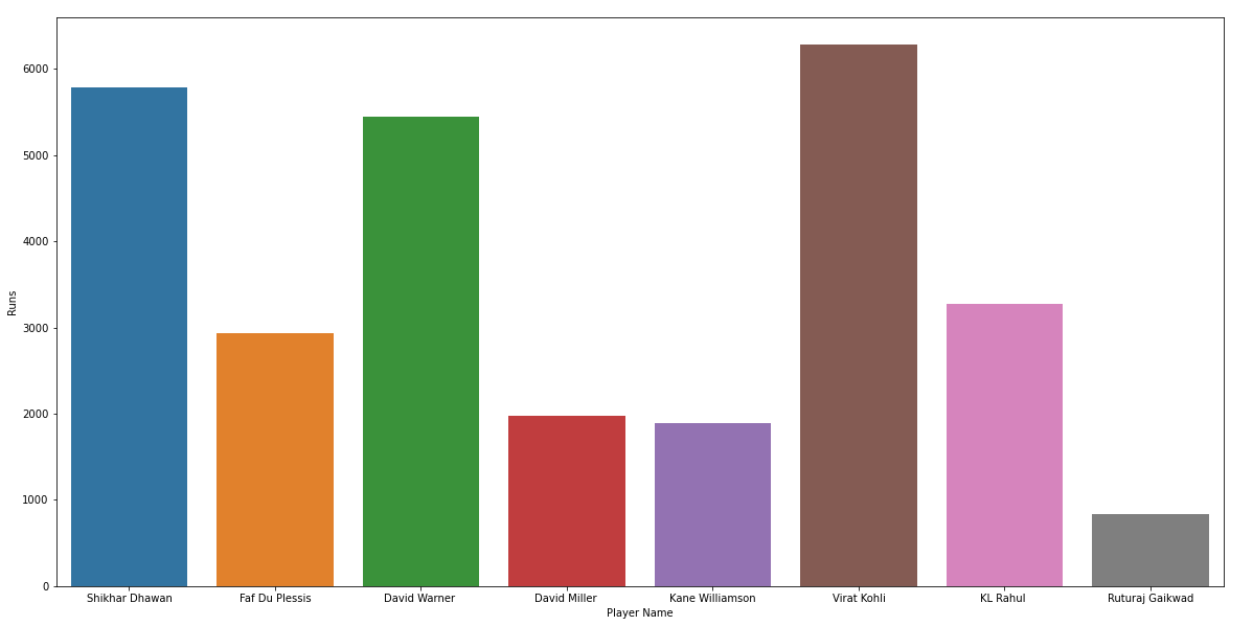
#PLot Shows each of the top batters.

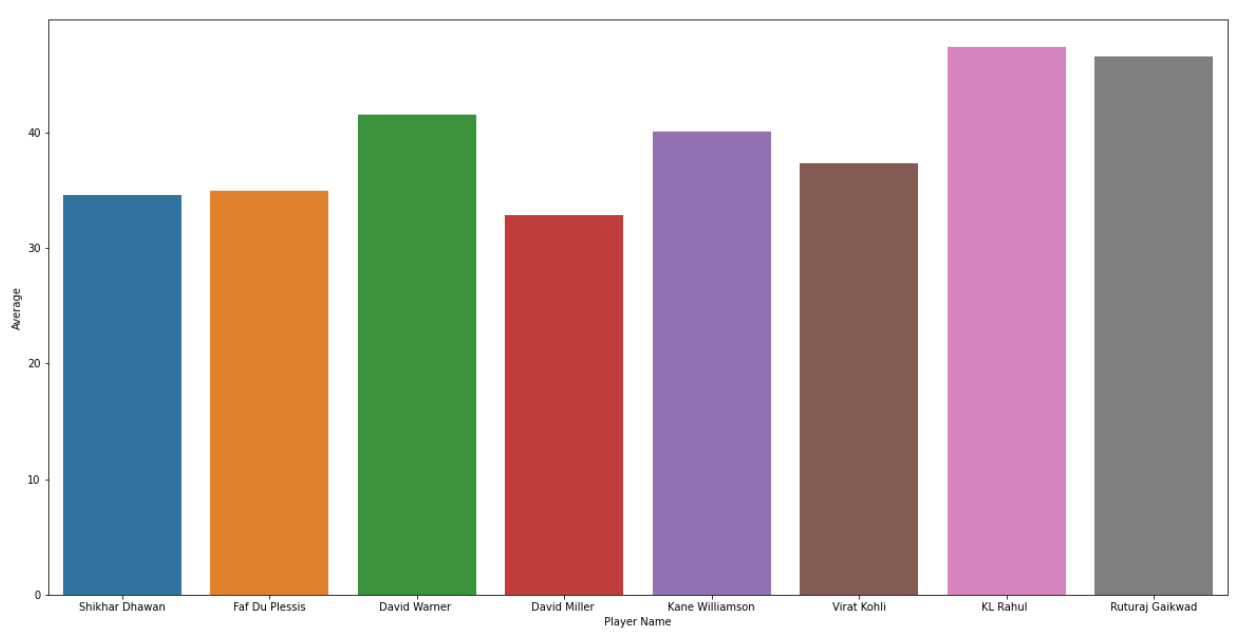
plt.figure(figsize=(20,10))

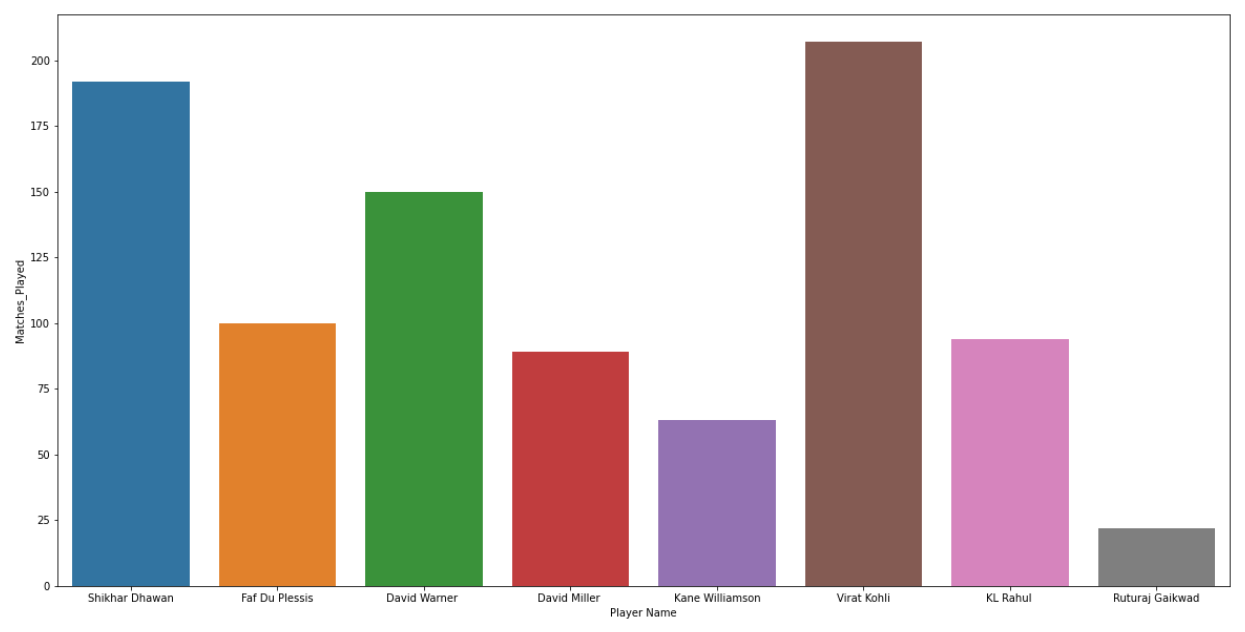
sns.barplot(x= 'Player Name', y = 'Matches\_Played', data = top\_batters)

**Output:**

****

****

****

****

**Code:**

import pandas as pd

#visualization of allrounders data

#PLot Shows each of the top allrounders.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Runs', data = top\_allrounders)

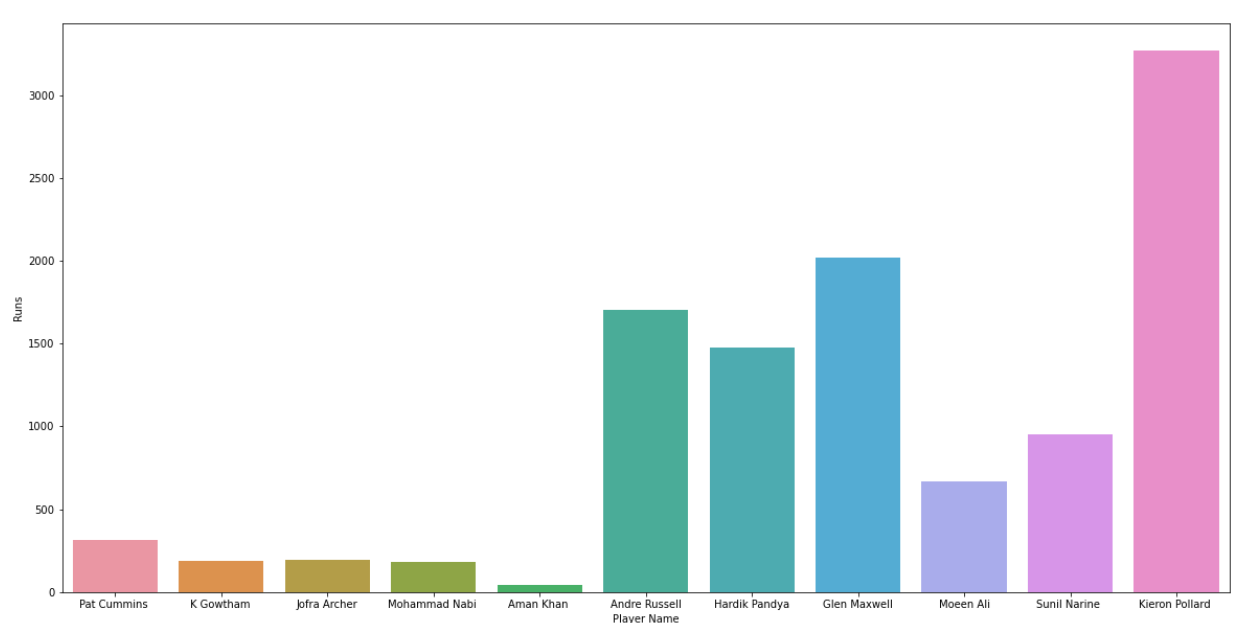
#visualization of allrounders data

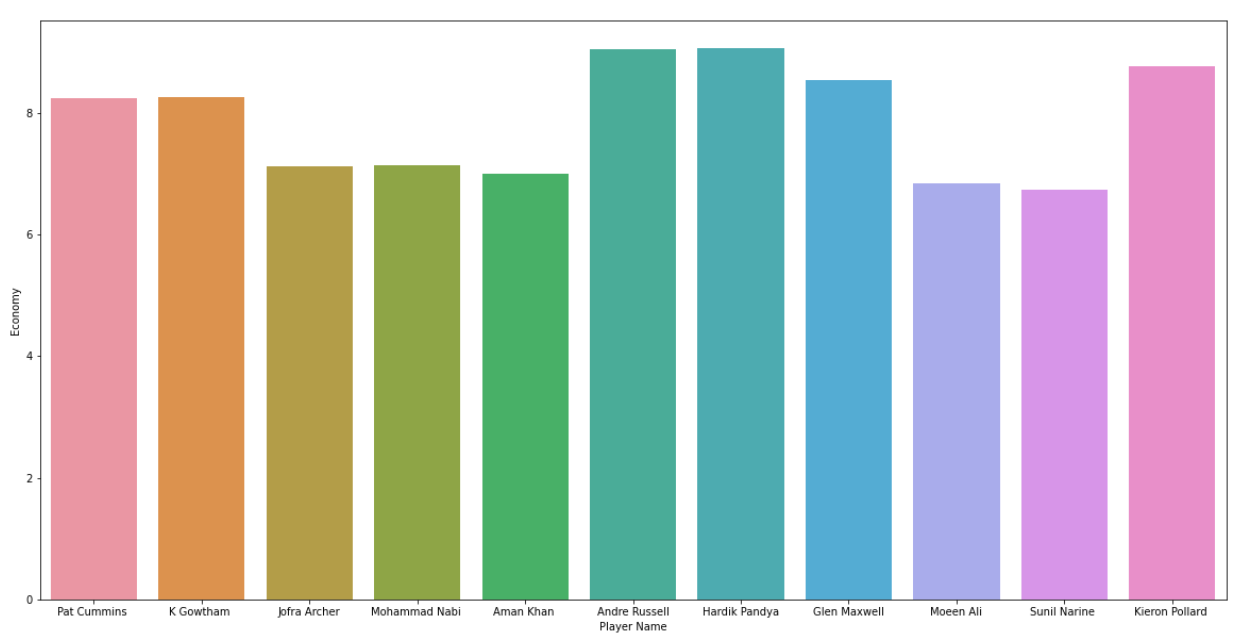
#PLot Shows each of the top allrounders.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Economy', data = top\_allrounders)

**Output:**

****

****

**Code:**

#visualization of keepers data

#PLot Shows each of the top keepers.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Average', data = top\_keepers)

#visualization of keepers data

#PLot Shows each of the top keepers.

plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Catches', data = top\_keepers)

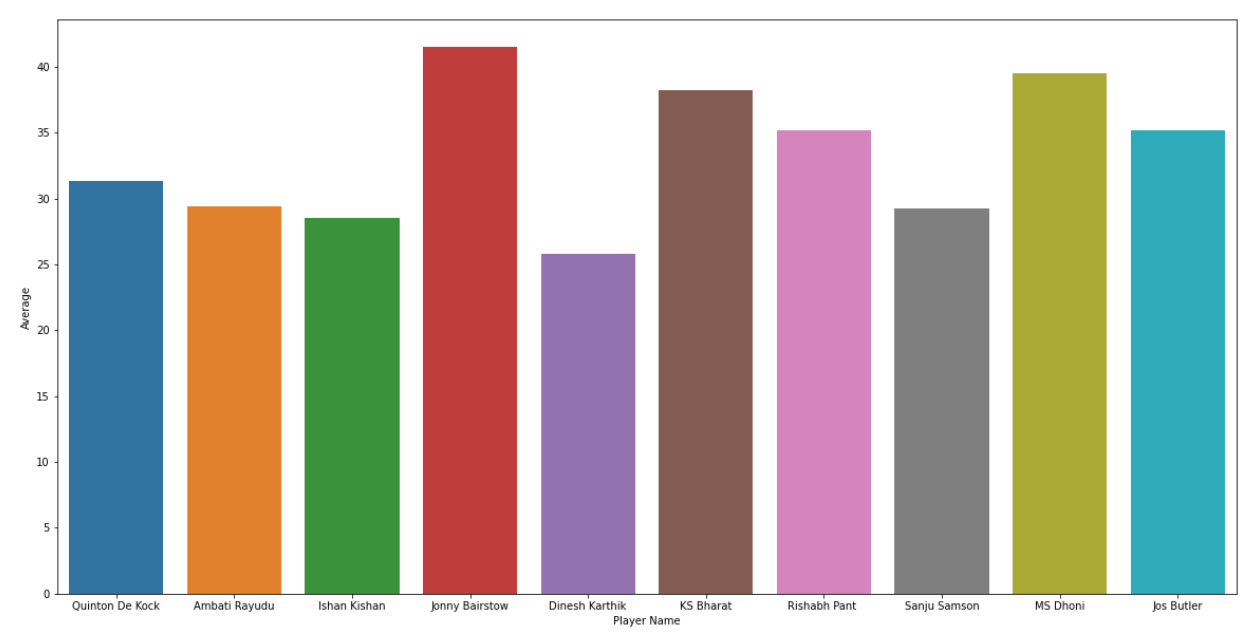
#visualization of keepers data

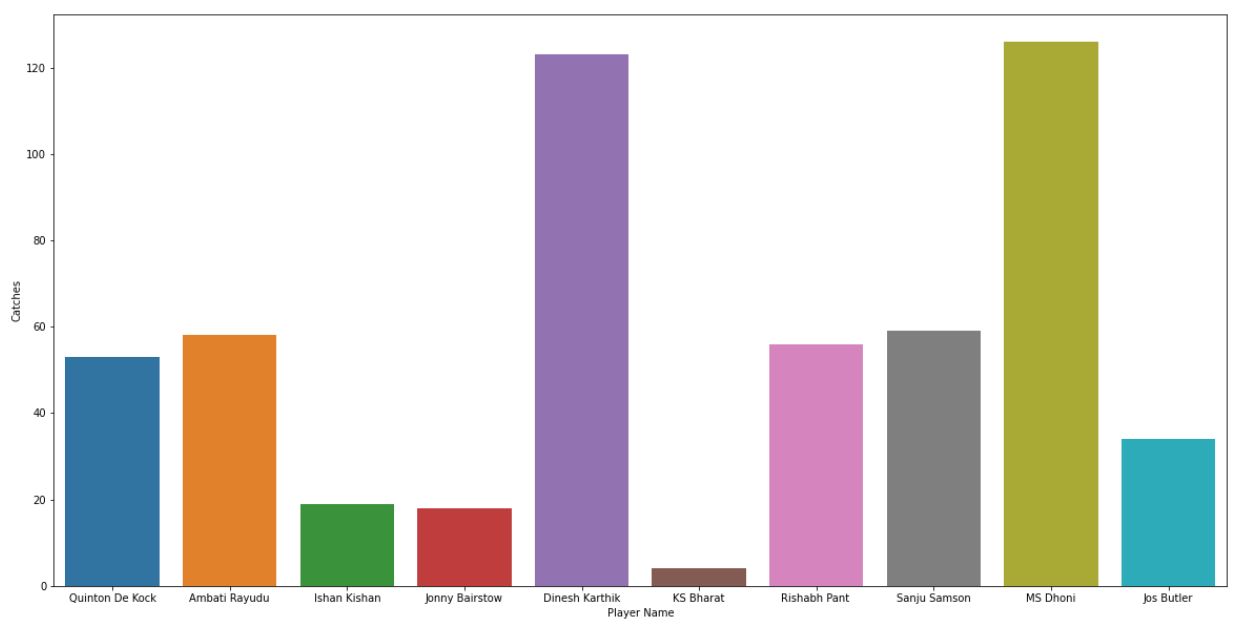
#PLot Shows each of the top keepers.

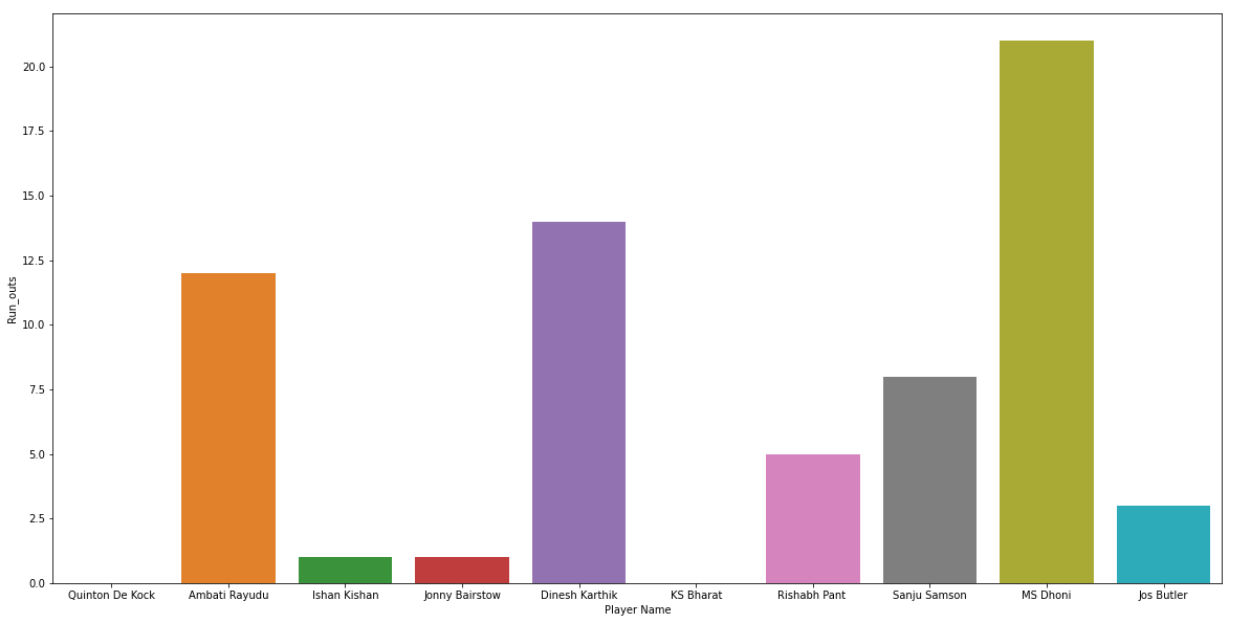
plt.figure(figsize=(20,10))

sns.barplot(x= 'Player Name', y = 'Run\_outs', data = top\_keepers)

**Output:**

****

****

****

**Code:**

#Batters for the Final 11 KL Rahul, Virat Kohli, David Warner

#here, we are storing the values of each player in a separate data frame to use for displaying using the barplot.

top\_batters.reset\_index(drop=True)

matches\_values = [top\_batters.iloc[6]['Matches\_Played'],top\_batters.iloc[2]['Matches\_Played'],top\_batters.iloc[5]['Matches\_Played']]

runs\_values = [top\_batters.iloc[6]['Runs'],top\_batters.iloc[2]['Runs'],top\_batters.iloc[5]['Runs']]

average\_values = [top\_batters.iloc[6]['Average'],top\_batters.iloc[2]['Average'],top\_batters.iloc[5]['Average']]

Strike\_rate\_values = [top\_batters.iloc[6]['Strike\_Rate'],top\_batters.iloc[2]['Strike\_Rate'],top\_batters.iloc[5]['Strike\_Rate']]

Labels = [ 'KL Rahul', "David Warner", "Virat Kohli"]

fig,axes =plt.subplots(2,2, figsize=(10,10))

axes[0][0].set\_title("Matches Played")

axes[0][1].set\_title("Runs in the IPL Career")

axes[1][0].set\_title("Strike\_Rate")

axes[1][1].set\_title("Average")

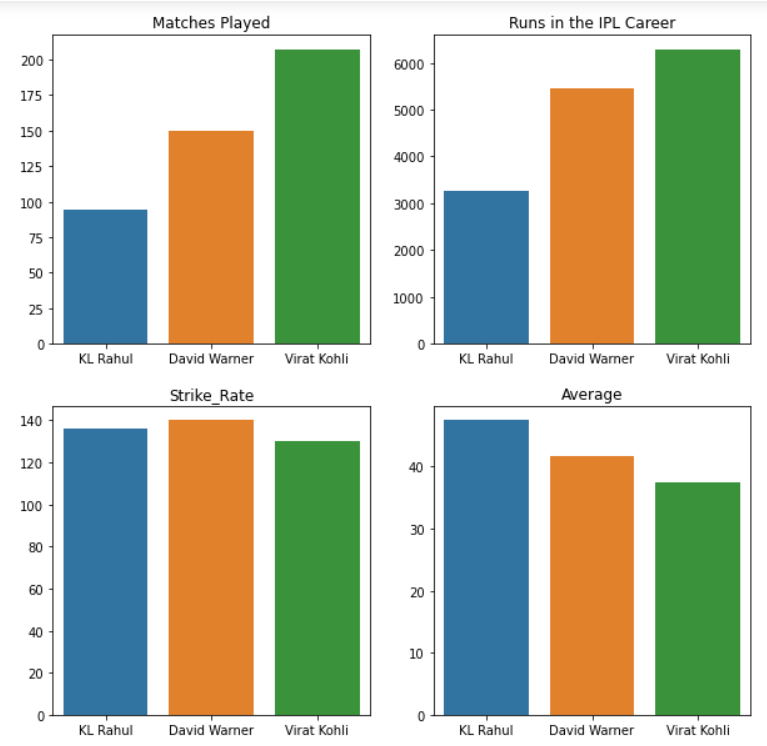
sns.barplot(x=Labels, y=matches\_values, ax=axes[0][0])

sns.barplot(x=Labels, y=runs\_values, ax=axes[0][1])

sns.barplot(x=Labels, y=Strike\_rate\_values,ax=axes[1][0])

sns.barplot(x=Labels, y=average\_values, ax=axes[1][1])

**Output:**

****

**Code:**

#Wicket Keeper For The Final 11- MS Dhoni

matches\_values = [top\_keepers.iloc[8]['Matches\_Played'],top\_keepers.iloc[8]['Runs']]

average\_values = [top\_keepers.iloc[8]['Average'],top\_keepers.iloc[8]['Strike\_Rate']]

keeping\_values = [top\_keepers.iloc[8]['Catches'],top\_keepers.iloc[8]['Stumps'],top\_keepers.iloc[8]['Run\_outs']]

label1= ['Matches','Runs']

label2= ['Average','Strike\_Rate']

label3= ['Catches','Stumps','Run-outs']

fig, axes = plt.subplots(1,3,figsize=(20,10))

axes[0].set\_title('Matches and Runs' )

axes[1].set\_title('Average and Strike Rate')

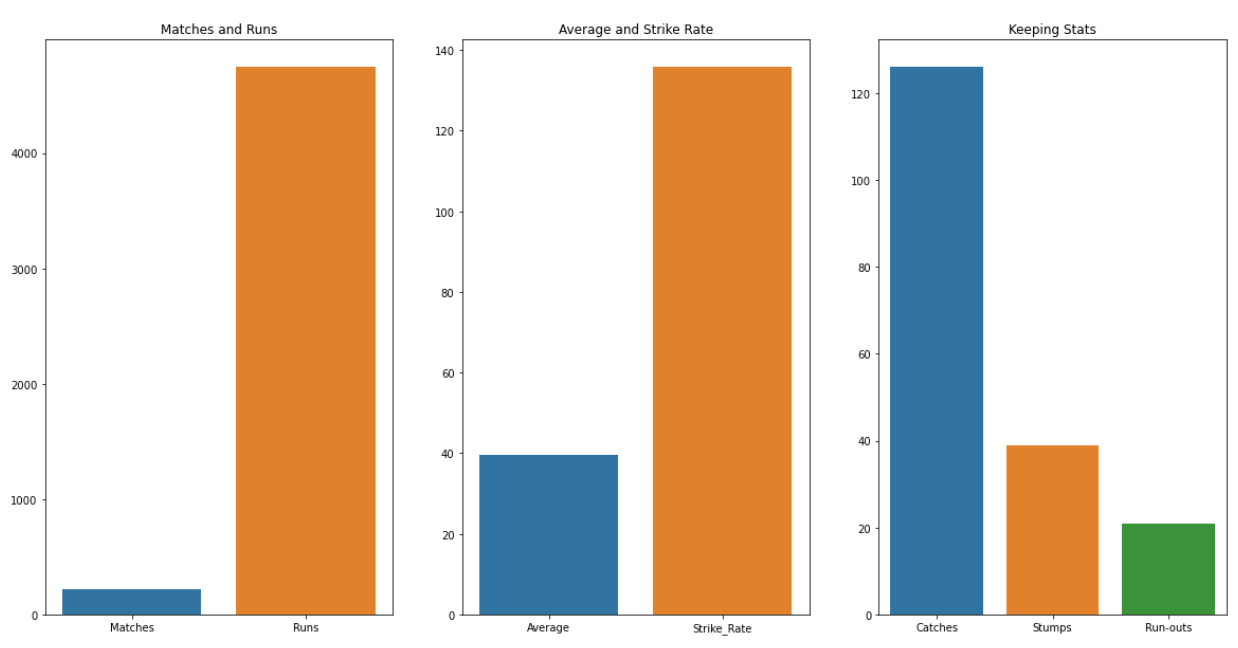
axes[2].set\_title('Keeping Stats')

sns.barplot(x=label1, y= matches\_values, ax= axes[0])

sns.barplot(x=label2, y= average\_values, ax= axes[1])

sns.barplot(x=label3, y= keeping\_values, ax= axes[2])

**Outputs:**

****

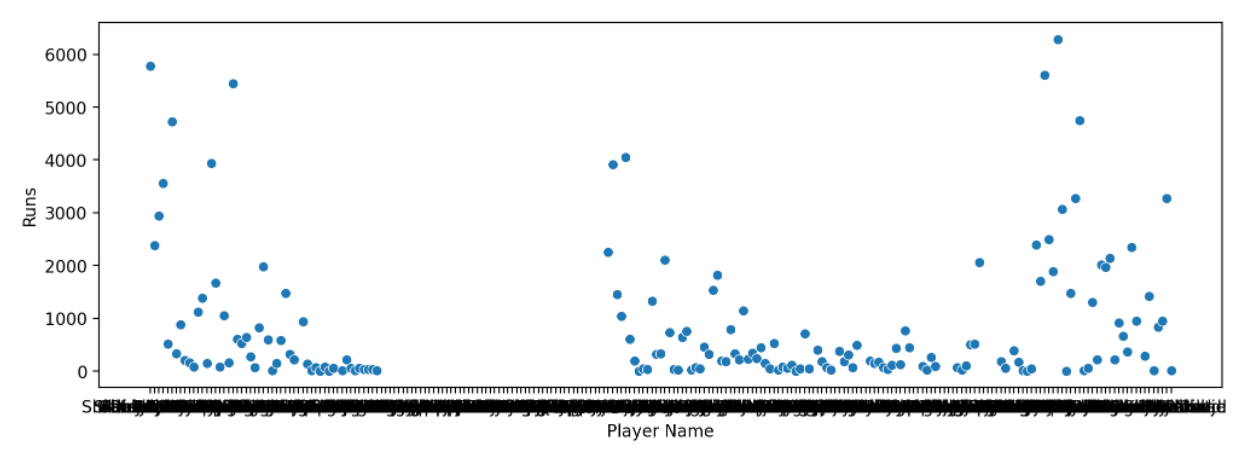
**Code:**

#Scatter plot

plt.figure(figsize=(12,4),dpi=200)

sns.scatterplot(x= 'Player Name', y = 'Runs', data = ipl)

**Output:**

****

**Code:**

#Line Plot for Capped\_Allrounders.

plt.plot(Capped\_Allrounders.Player\_Name,Capped\_Allrounders.Bowling\_average,label="Bowling\_average")

plt.plot(Capped\_Allrounders.Player\_Name,Capped\_Allrounders.Matches\_Played,label="Matches\_Played")

plt.plot(Capped\_Allrounders.Player\_Name,Capped\_Allrounders.Bowling\_Strike\_Rate,label="Bowling\_Strike\_Rate")

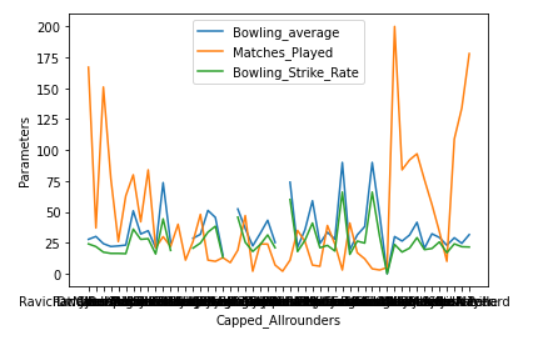
plt.xlabel('Capped\_Allrounders')

plt.ylabel('Parameters')

plt.legend()

plt.show()

**Output:**



**Conclusion and Future Scope:**

# Final Playing 11 on above analysis will be

#1.KL Rahul

#2.David Warner

#3.Virat Kohli

#4.Andre Russell

#5.Sunil Narine

#6.Hardik Pandya

#7.Ms Dhoni

#8.Yuzvendra Chahal

#9.Jasprit Bumrah

#10.Nathan Coulter-Nile

#11.Kagiso Rabada

The results show that the problem of match result

prediction can be solved and a mathematical model can be created for prediction of the results of the matches prior to the match based on the knowledge of past matches, playing eleven and the toss result. In this work 5 features of IPL career and 5 features of International T20 Career have been taken into consideration for both batsmen and bowlers but in future work more features can be created and taken into account. The training data can be made larger in the future work for better model learning and classification. Various other data analytics techniques can also be used for improving the accuracy of the model in future work

**References:**

* <https://www.youtube.com/watch?v=vC7xtwdE5-Q>
* <https://www.kaggle.com/kalilurrahman/ipl-player-auction-data-analysis>
* <https://www.w3schools.com/python/matplotlib_pyplot.asp>